When Will Ray-Tracing Replace Rasterization?

Larry Seiler
ATI Research, Inc.
LSeiler@ati.com

Ray-Tracing Assertions

- **It produces great lighting effects**
  - Also true for volumetric ray-tracing
- **It can now run on standard GPUs**
  - It will work even better in the future
  - Also true for volumetric ray-tracing
- **So ray-tracing will soon replace rasterization?**

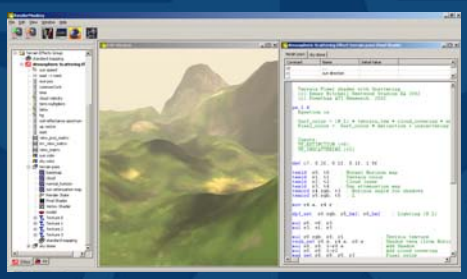
Stop Rasterizing? Never!

- **When did we replace**
  - Cobol? Fortran? VGA? MS/DOS?
  - OK, three of these are going away, hopefully…
- **We still use a lot of "old methods"**
  - 2D area fill, bitblt (2D copy), line drawing…
- **But there are more fundamental reasons to keep rasterizing**

Outline

- **Examples of rasterization effects**
- **Reasons to keep rasterizing**
  - Memory bandwidth
  - GPUs/VPUs vs. CPUs
  - Algorithmic advantages
- **Future directions**
- **Real-time rasterization demos**

Atmospheric Scattering



Non-Photorealistic Rendering

Motion Blur

Is it Ray-Traced, or...?

False Reasons to Stop Rasterizing

- **To get realistic lighting effects**
  - The goal is information, not just effects
- **To allow a simple unified model**
  - Instead, choose the best technique per effect
- **Ray-tracing is now fast enough**
  - But rasterization is often a lot faster
  - Moore's law applies to both

Memory Bandwidth

- **Memory density and cost per bit have followed Moore's law**
  - This & bin sort is why Sutherland was wrong: depth buffers became cheaper and faster
- **Memory bandwidth per data pin hasn't**
  - Wider data busses help, but not enough
  - Rasterization provides good memory locality for efficient caching & burst accesses

GPUs/VPUs are Stream Processors, *NOT* CPUs

- **Both are general purpose, but...**
- **GPU: Massive SIMD parallelism**
  - SIMD-like conditionals (MIMD is costly)
- **GPU: Streaming reads and writes**
  - GPU: 5x CPU latency on memory-to-memory
  - >20x latency on load A0/store A1/load A1
  - Coherency is inherently very costly

Algorithmic Advantages

- **Efficient depth culling on pixels**
  - Hierarchical depth tests save shading work
  - Can occlusion cull triangles as well
- **Simple layering effects**
  - Compute lighting and shadows separately
  - Compute an image, then use it as a texture
  - Ray-trace where appropriate

Future Directions

- **Anti-aliased lighting**
  - See Jim Blinn's 1998 SIGGRAPH keynote
- **Surfels/splatting/randomized Z-buffer**
  - Pfister (2000), Zwicker (2001), Wand (2001)
- **Eliminating unnecessary work**
  - Jones, et. al. "Shader Maps" (2001 sketch)
- **Even more this year…**

Lighting Effect Demos: Real Time Rasterization

- **"Rendering with Natural Light" by Paul Debevec**
  - From SIGGRAPH 1998 Electronic Theater
  - See also Debevec, "Rendering Synthetic Objects into Real Scenes", SIGGRAPH 1998
- **"Pipe Dream" by ANIMUSIC**
  - From SIGGRAPH 2001 Electronic Theater
  - See also www.animusic.com

Conclusions

- **Ray-tracing and rasterization can co-exist, since each has advantages**
- **Rasterization and depth buffering allow merging many techniques**
- **"Rumors of rasterization's death are greatly exaggerated"**